Remarks

Applicant respectfully requests reconsideration of this application as amended. Claims 1-30 are currently pending.

All existing claims have been cancelled without prejudice. Claims 31-50 have been added.

Therefore, claims 31-50 are hereby presented for examination.

Objections to the Drawings

The Examiner has objected to Figure 4 because of the inclusion of reference elements 425 and 440, which were inadvertently omitted from the Specification. Paragraph 0062 of the Specification has been modified to identify elements 425 and 440.

It is submitted that the amendments to the Specification fully address the objections to the drawings raised by the Examiner, and that no further modifications are needed.

35 U.S.C. § 112

The Examiner rejected claim 22 under 35 USC § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter that the Applicant regards as the invention.

Claim 22 has been cancelled, and it is submitted that the antecedent basis issue raised by the Examiner is no longer relevant to the current claims presented in this application.

35 U.S.C. § 102

Eberhard

Claims 1-2, 8, 11-12, 19, 22 and 28 are rejected under 35 U.S.C. §102(b) as being anticipated by Eberhard U.S. Pat. Pub. No. 2002/0078264 ("*Eberhard*").

The rejected claims have been cancelled. It is submitted that that *Eberhard* does not teach or suggest the elements of the current claims.

Claim 31 reads as follows:

31. A processor comprising:
an execution unit, the execution unit to execute instruction; and
a buffer to store data regarding a plurality of instructions executed by the execution unit, wherein the data stored for each executed instruction includes an instruction address and an effective address for the executed instruction;
wherein the storage of data in the buffer is to be halted upon the occurrence of an event, and wherein the processor is to determine a relationship between a set of executed instructions based on the stored data for the set of executed instructions in the buffer.

Thus, claim 31 provides a buffer to store data regarding a plurality of instruction loads executed by the processor, the data stored for each executed instruction including an instruction address and an effective address for the executed instruction. Further, the processor is to determine a relationship between a set of executed instructions based on the stored data for the set of executed instructions in the buffer. It is respectfully submitted that *Eberhard* does not teach or reasonably suggest these elements of the claims.

Eberhard provides for a system and method for capturing and storing trace signals. In this system, a memory controller used to manage a memory interface includes a trace array to accumulate trace data signals that will be sent to the main store. The trace data may be moved to the main store when the trace array is full, or when a storage request queue is empty.

However, it is not clear what type of data is collected in the system described in *Eberhard* or how the data may be utilized because the reference never mentions this. In the background, *Eberhard* mentions that processor architectures often provide mechanisms for "capturing instruction streams to aid performance analysis", and that there is also a need to provide for "capture of processor bus and I/O bus behavior", and that having "detailed and accurate information about instruction frequency and sequences" is important for developing designs having high performance. (*Eberhard*, ¶0004) However, this says nothing about what data might be collected, or how this data might be used.

With regard to the alleged invention, *Eberhard* indicates that one of the objects is "to provide a trace system and method which provides for capture of processor bus and I/O bus behavior". (*Eberhard*, ¶0009) This provides no guidance as to what type of data is being collected. *Eberhard* generally simply refers to "trace data" or "trace data signals", as in "trace data signals are captured and driven to a trace array for storage" (*Eberhard*, ¶0015) without any explanation as to what type of data is being collected. *Eberhard* further indicates that "a method and system is provided for capturing instruction sequence and instruction frequency data in a processor's main store--these are signals which are useful for hardware or software debug and performance monitoring." (*Eberhard*, ¶0020) This implies that the data collected relates to instruction sequence and

instruction frequency data, with no mention of any address data. Similarly Figure 1 contains a trace array 112 for "signal values", but what type of signal values might be contained in the array is not specified. "Embedded within memory controller 100 is trace array 112 and its corresponding trace array control logic 113. Trace array 112 is a simple storage array N entries deep by M bits wide <u>which is used as temporary storage for signal values</u> which are to be stored to main store 110." (*Eberhard*, ¶0029)

Eberhard does mention addresses, but what are described are <u>not</u> address data elements. Instead, what *Eberhard* discusses is <u>the addresses at which collected data is stored</u>, which is a completely different concept. "Which signals to record, when recording should begin, and <u>which addresses in main store are used to store the signal values</u> are selectable via program control." (*Eberhard*, ¶0020) (emphasis added) The reference further describes how storage addresses are made to determine whether there is space available for storage. "When the Nth location of the storage array has been updated with data, the array update control is disabled. The current offset address is compared to the addresses provided in the trace starting address register and space size register. <u>This comparison is performed to determine if enough memory remains within a predefined space in main store to store the contents of the storage array</u>." (*Eberhard*, ¶0025) (emphasis added) "In steps 126 and 128, once all available entries within trace array 112 contain captured signal values, signal capture is disabled and, in step 130, trace array control logic 113 <u>compares the size of trace array 112 and the address stored in current trace address register 108 to the address stored in trace end address register 109</u>. If the store request exceeds the capacity of the address space assigned to trace, in step 132 attention signal 115 is activated, indicating that the address space reserved in main store 110 to record trace data has been exhausted." (*Eberhard*, ¶0031) (emphasis added)

Thus, *Eberhard* is a reference that relates to a system for capturing and storing trace data signals. However, there is nothing in *Eberhard* that explains what type of data is being stored, or how the data is utilized. It is thus submitted that *Eberhard* does not teach or suggest a buffer to store data regarding a plurality of instruction loads executed by the processor, where the data stored for each executed instruction includes an instruction address and an effective address for the executed instruction and where the processor is to determine a relationship between a set of executed instructions based on the stored data for the set of executed instructions in the buffer.

35 U.S.C. § 103

Talcott, et al. in view of Brock, et. al.

Claims 22-23, 25-27, and 29-30 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Talcott, et al., U.S. Pat. Pub. No. 2003/0188226 ("*Talcott*") in view of Brock, et al., U.S. Pat. No. 6,601,149 of ("*Brock*").

The rejected claims have been cancelled. It is submitted that that *Talcott* and *Brock*, together or in combination with each other or with *Eberhard*, do not teach or suggest the elements of the current claims. In particular, it is submitted that *Talcott* and *Brock* do not teach or suggest the elements of claim 31 to provide a buffer to store data regarding a plurality of instruction loads executed by a processor, the data stored for each executed instruction including an instruction address and an effective address for the executed instruction, wherein the processor is to determine a relationship between a set of executed instructions based on the stored data for the set of executed instructions in the buffer.

Talcott describes a sample mechanism which is intended to allow software to specify a property or properties that characterize samples of interest, thereby allowing a sampling mechanism to discard or ignore uninteresting samples. In this system, certain data regarding instructions is collected. "Referring to FIG. 1, processor 100 includes sampling mechanism 102. This sampling mechanism 102 is provided to collect detailed information about individual instruction executions." (*Talcott*, ¶0017) However, there is again a question regarding what data is actually being collected and how it is being used. As described in *Talcott*:

> [0019] The sampling mechanism 102 collects detailed information about individual instruction executions. If a sampled instruction meets certain criteria, the instruction becomes a reporting candidate. When the sampling mode is enabled, instructions are selected randomly by the processor 100 (via, e.g., a linear feedback shift register) as they are fetched. An instruction history is created for the selected instruction. The instruction history is made up of a vector of information including such things as events induced by the sample instruction and various associated latencies. When all events for the sample instruction have been generated (e.g., after the instruction retires or aborts), the vector of events gathered by the instruction history is compared with a user supplied vector, which indicates the events of interest.

(*Talcott*, ¶0019) Thus, detailed information regarding individual instruction executions is collected, and an instruction history is created for a selected instruction, which may include such things as events induced by the sample instruction and various associated latencies.

Figures 2A and 2B show a flow chart of the operation of the sampling mechanism. Within this discussion, there is an indication that instruction information is

captured, element 230. "If the fetched instruction is a valid instruction, then instruction information is captured at step 230. The instruction information includes, for example, the program counter (PC) of the instruction as well as privileged information and context information of the instruction." (*Talcott*, ¶0022) Thus, the collected information may include a program counter of the instruction and privileged information and context information of the instruction.

Figure 3 of *Talcott* then provides a description of registers, particularly SIH register set 304. (Element 302 is not relevant as it describes the sample selection criteria registers, which may include "an interesting event register, a PC range register, a latency mask register, and a privileged, nonprivileged register". (*Talcott*, ¶0027)) Element 304 is described as follows:

> [0029] The set of SIH registers 304 include a plurality of registers. More specifically, the set of SIH registers 304 include an events register value, a PC register, a branch target address register, an effective memory address register, a latency register, a number in issue bundle register, a number in retire bundle register, a privileged register, a branch history register, and a number in fetch bundle register.

(*Talcott*, ¶0029) Thus, the data collected may include "a branch target address register" and "an effective memory address register". However, there is no indication that the data collected is utilized to determine a relationship between a set of instructions, or for that matter how any of the collected data may be applied other than to generally understand the behavior of a program.

Brock concerns a system for monitoring memory transactions in a data processing system. The system is intended to define a set of memory transaction attributes, and to

detect memory transactions that match the set of memory transaction attributes. (e.g., *Brock*, col. 2, lines 18-25) Upon detecting memory transactions, the system is intended to graphically display the number of transactions occurring during a specified duration. (e.g., *Brock*, col. 2, lines 25-27) For example, Figure 8 of *Brock* illustrates a memory transaction histogram generated by the claimed monitoring system. The histogram 800 indicates a count number and vertical bar to describe each "grain" of memory.

Brock further describes the use of the graphical data to examine memory issues. "In the preferred embodiment, monitoring system 103 is configured to present the user with graphical representations of transactions monitored on switch 109. In addition, monitoring system 103 provides the user with selectable inputs that permit the user to visualize various types of transactions and to determine the regions in physical memory corresponding to the monitored transactions. With the graphical interface provided by monitoring system 103, a user can gather empirical memory access information to discover any memory performance inefficiencies or abnormalities that may exist within system 100." (*Brock*, col. 7, lines 45-55)

Thus, what *Brock* is describing is a system for defining certain memory transaction attributes and for detecting memory transactions that meet the attributes. Having detected the transactions, *Brock* provides for graphically displaying the detected memory transactions. While there are other differences, *Brock* specifically does not teach or suggest the elements of claim 31 to provide a buffer to store data regarding a plurality of instruction loads executed by a processor, the data stored for each executed instruction including an instruction address and an effective address for the executed instruction, wherein the processor is to determine a relationship between a set of executed instructions based on the stored data for the set of executed instructions in the buffer.

It is respectfully submitted that none of the cited references teach or suggest the elements of claim 31, and the claim is thus allowable. It is submitted that the arguments provided above also apply to independent claims 39 and 46, and such claims are also allowable. The remaining claims are dependent claims, and are allowable as being dependent on the allowable base claims.

Invitation for a Telephone Interview

The Examiner is requested to call the undersigned at (503) 439-8778 if there remains any issue with allowance of the case.

Request for an Extension of Time

Applicant respectfully petitions for an extension of time to respond to the outstanding Office Action pursuant to 37 C.F.R. § 1.136(a) should one be necessary. Please charge our Deposit Account No. 02-2666 to cover the necessary fee under 37 C.F.R. § 1.17(a) for such an extension.

Charge our Deposit Account

Please charge any shortage to our Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 3/19/2007

/Mark C. Van Ness/
Mark C. Van Ness
Reg. No. 39,965

12400 Wilshire Boulevard
7th Floor
Los Angeles, California 90025-1030
(503) 439-8778